LA-UR-76-2656

c-1

IMPACT OF SCALAR PERFORMANCE ON VECTOR AND PARALLEL PROCESSORS

Lawrence E. Rudsinski
William J. Worlton

THE IMPACT OF SCALAR PERFORMANCE ON VECTOR AND PARALLEL PROCESSORS*

by

L. Rudsinski and J. Worlton
UNIVERSITY OF CALIFORNIA
Los Alamos Scientific Laboratory
Los Alamos, N. M. 87545
FTS Number: 843-7158
Phone Number: (505) 667-7158

ABSTRACT

The large scientific programs that require the computational power of a vector or parallel computer consume many hours of computer time for their completion. It is estimated that about 15 percent of this computational work is done in scalar mode, and the scalar performance can therefore be a critical component in the overall performance of a vector or parallel processor. Using heuristic models for both vector and parallel performance, this paper analyzes the overall performance of a supercomputer as a function of both scalar performance and the fraction of results generated in vector or parallel mode.

SUMMARY

1. INTRODUCTION

When considering the use of a vector or parallel computer, we must assume a nontrivial amount of computational work will be done in scalar mode, especially in the large-scale calculations that are typical of scientific research. At Los Alamos Scientific Laboratory (LASL) to qualify for such a computer, a program must consume at least one Control Data Corporation (CDC) 7600 hour per run and require more than one such run per week. These programs have evolved over many years and become very large, complex and unstructured. If we look at a typical subroutine from one of these codes, we find that the Fortran appears to be sequential and little if any use could be made of the vector or parallel features of the new machines. These programs were developed on sequential computers and the codes reflect this. Rewriting a representative production program to exploit the potential parallelism in both the algorithms and the supercomputer's architecture will require several man-years. Of course, those sections of the program that dominate the total computational work should be vectorized. The point is that in a typical implementation of these codes on a vector computer large sections of code constituting a few percent of the computational work will continue to be implemented in scalar formulation.

*This work performed under the auspices of the U. S. Energy Research and Development Administration.

During the execution of any given problem, production codes will have substantial interaction with the operating system, if for no other reason than to perform I/O. We have found at LASL that a typical production program spends 5-15% of its compute time in the operating system during its execution. Operating systems are sequential by nature, so even in the most idealistic case where all the algorithms in a code are parallel, scalar mode is inescapable. Thus, when considering the usage of a vector or parallel computer, we must assume a nontrivial amount of computational work will be done in scalar mode. These observations are consistent with Amdahl's findings [1] that, under the best conditions, 15-30% of the compute time will be spent in scalar mode. This paper will address the question of how important this scalar component is with respect to overall performance. We will show that it is extremely important.

2. A HEURISTIC MODEL OF SCALAR-VECTOR PERFORMANCE

We now develop a "heuristic model" employing a high level of abstraction; although quantitative results may not be precise for particular architectural implementations, the model gives correct qualitative insights into the impact of scalar performance on scalar-vector processors.

We assume a vector start-up time, S, is required for each vector operation, and that each vector result is generated in time t_v, so the total time to complete a vector operation of length L is given by

$$T_v = S + L*t_v \qquad (1)$$

The mean time per vector result is then

$$T_v/L = S/L + t_v \qquad (2)$$

We assume a scalar result generation time of t_s, and the fractions of results in vector mode and scalar mode are F and (1-F), respectively. The time per result in a mixed scalar-vector calculation is given by

$$\langle T \rangle = F*(S/L + t_v) + (1-F)*t_s \qquad (3)$$

and the execution bandwidth, B, in results per unit time is given by

$$B = 1/\langle T \rangle = 1/(F*(S/L + t_v) + (1-F)*t_s) \qquad (4)$$

We note that while this is only a heuristic model for scalar-vector processors in general, it represents memory-to-memory (MM) vector processors quite accurately, and it can be used as a "refined" MM model.

A strictly scalar processor would have an execution bandwidth given by

$$B_S = 1/t_S \qquad (5)$$

and the relative performance of a scalar-vector processor tc that of a strictly scalar processor is

$$R = B/B_S = 1/(F*(S/L + t_V)/t_S + (1-F)) \qquad (6)$$

For a scalar-vector processor operating on infinite-length vectors at infinite speed ($t_V = 0$), equation (5) becomes just $R = 1/(1-F)$, which is the upper bound that can be achieved by a scalar-vector processor relative to a scalar processor having the same scalar execution bandwidth.

Figure 1 shows the performance bounds for three different ratios of scalar performance in the machines being compared. The values on the ordinate are the ratio of the performance of the scalar-vector machine to the existing scalar machine. The first scale indicates that both machines perform scalar operations equally well. Note that even for an infinitely fast vector processor, the performance gain from 50% vector work is at most a factor of two; the performance gain from 75% vector work is at most a factor of four; etc. This limitation on the performance gain is entirely due to the scalar work that remains after the time to complete the vector work is set to zero.

The second scale in Figure 1 illustrates the situation where the scalar speed of the scalar-vector computer is significantly slower than that of the existing machine. In particular, this ordinate displays the case where the scalar mode of the new machine is one fourth that of the existing machine. Note that we must have 75% vector results just to "catch up" with the performance of the existing scalar machine. Because we are also assuming infinite vector speed, in a real machine with these scalar characteristics vector results will have to be somewhat higher than 75% to "break even." This illustrates the severe penalty of incorporating a slow scalar processor into a scalar-vector architecture.

The third scale in Figure 1 shows the effect of speeding up the scalar processor in a scalar-vector architecture by a factor of two. Clearly, the effect of scalar performance on scalar-vector architecture is profound when the fraction of vector results is in the ranges noted in Section 1.

3. A HEURISTIC MODEL OF SCALAR-PARALLEL PERFORMANCE

The effects of scalar performance on parallel architectures are similar to those derived in Section 2; here "scalar" performance is defined to mean strictly sequential performance in which only one processing element (PE) is active.

If we assume that either one PE is active or all PEs are active, and that the fraction of results generated in the parallel mode and scalar mode are given by F and (1-F) respectively, then the mean time per result in a parallel processor having N PEs is given by

$$\langle T_p \rangle = F*(1/N) + (1-F) \qquad (7)$$

and the execution bandwidth is given by

$$B_p = 1/(F*(1/N) + (1-F)) \qquad (8)$$

If we now let N increase without bound, i.e., we assume an infinite number of PEs are active in parallel mode, we obtain the same limit noted in Section 2, i.e.,

$$B_p = 1/(1-F). \qquad (9)$$

Thus, the effects of strictly sequential work on parallel processor performance are the same as the effects of scalar performance on vector processors.

4. QUANTITATIVE RESULTS FROM THE EVALUATION OF THE CRAY-1

A practical application of these concepts recently occurred at LASL in the need to evaluate a scalar-vector computer architecture, the CRAY-1. LASL performed a six-month evaluation (April-September 1976) of the CRAY-1 computer to determine if this machine met the minimum performance standards set forth by LASL and the Energy Research and Development Administration (ERDA) to qualify it for further consideration for procurement. A major share of this evaluation effort was spent on scalar performance.

A preliminary test of seventy lines of Fortran code was selected from an equation of state subroutine. This module was written efficiently in assembly language in scalar mode for both the CDC 7600 and CRAY-1. The ratio of execution times of the 7600 to CRAY was 2.55. The final scalar test was a statistical selection of 22 small (2-10 Fortran statements) modules. Again the modules were programmed efficiently in assembly language for both machines and carefully timed. The average ratio of the 7600 to the CRAY times was 2.05, with no module dropping below a ratio of 2.

Given the significantly faster scalar performance of the CRAY-1 as compared to the 7600, this scalar-vector processor can be used in a mixed scalar-vector computing environment to exploit vector processing without the penalty of a large scalar degradation.

5. CONCLUSION

Scalar mode plays a critical role in almost any meaningful calculation. It is so important that if the scalar performance of new supercomputers cannot at least compete with the scalar mode of existing computers, the new machines probably do not warrant further consideration for procurement by users of such machines. On the other hand, if the scalar performance of the new machines is at least twice as fast as existing machines, then the programs that qualify can be moved directly to these machines with minimal effort and an immediate increase in throughput will be realized. This then allows a gradual modification of these codes to exploit both parallelism in algorithms and the new hardware.

Because scalar mode is so important, LASL went to great pains to evaluate the scalar performance of the CRAY-1 during the six-month evaluation period. Both the preliminary scalar test and the final scalar evaluation showed that the CRAY-1 scalar performance was greater than a factor of two over the CDC 7600 scalar performance.

REFERENCE

1. Amdahl, Gene M.,"Validity of the single-processor approach to achieving large-scale computing requirements," Computer Design, Vol. 6, No. 12, December 1967, pp. 39,40.

ACKNOWLEDGMENT

The authors wish to express appreciation to Bill Buzbee for his review of this paper and his suggestions for its improvement.

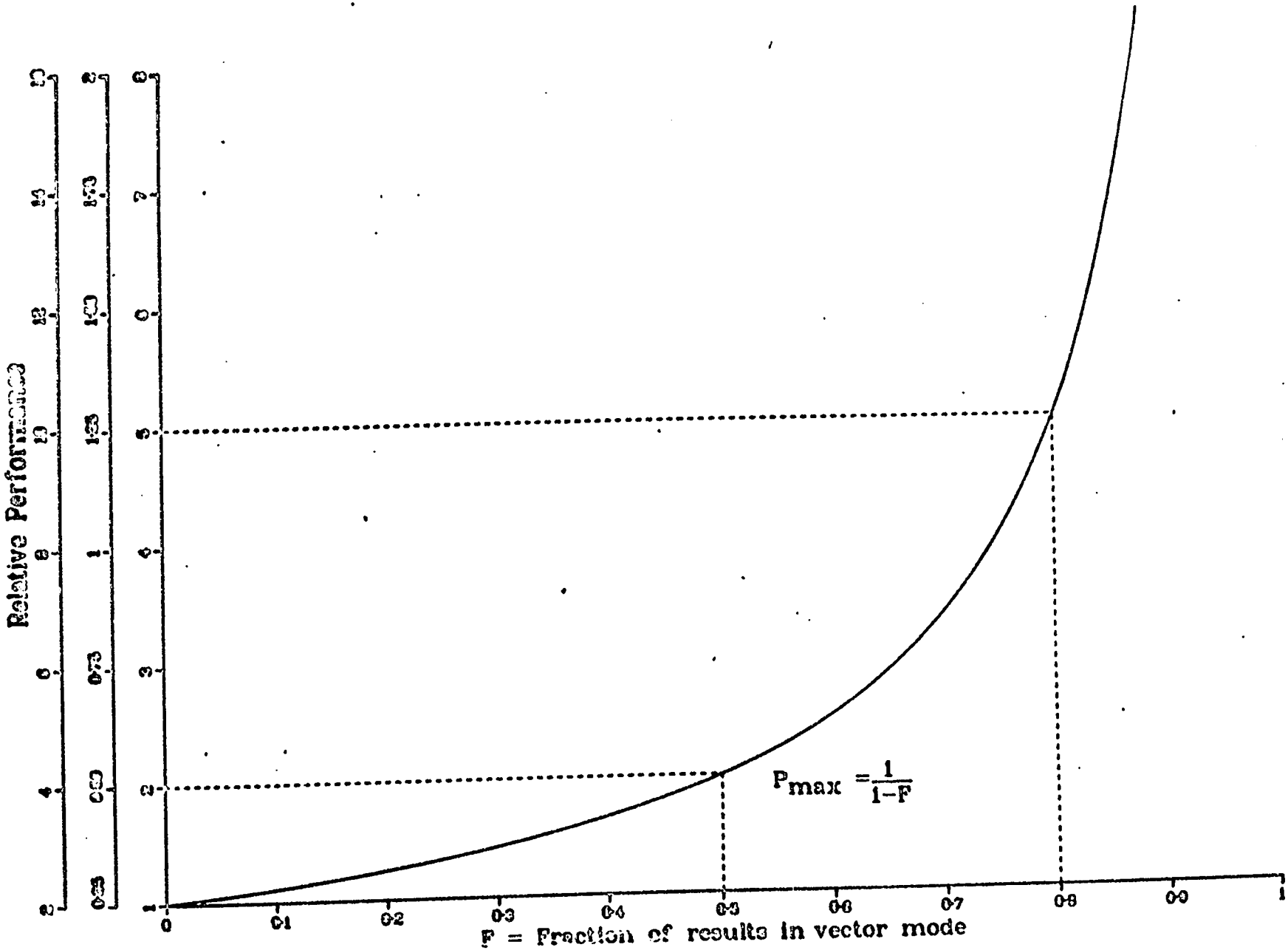

Figure 1. Maximum Performance Gain from Vectorization